IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

BUTTS, Michael

Serial No.: 10/669,095

Filed: September 23, 2003

For: LOGIC MULTIPROCESSOR FOR FPGA IMPLEMENTATION

Group Art Unit: 2825

Examiner: Yelena Rossoshek

AMENDMENT AND RESPONSE

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Office Action mailed April 19, 2006, please enter the following amendments.

IN THE CLAIMS

Please amend the following claims:

1. – 7. (Cancelled)

8. (Currently Amended) A method for implementing a design verification system into at least one programmable logic device so that a user design to be verified can be implemented therein, comprising:

mapping the user design into operations for execution;

partitioning each of said operations into processor types suitable for each of said operations;

ordering each of said processor types according to connectivity of each of said processor types;

scheduling communications between each of said processor types; and

programming said processor types into said at least one programmable logic device such that said at least one programmable logic is programmed to implement said processor types.

9. (Original) The method of claim 8 wherein said processor types comprise logic processors, macro processors, memory processors and general purpose processors.

10. (Previously Presented) The method of claim 9 wherein said memory processor comprises;

an instruction memory;

a memory functional unit that can store data; and

a register file controlled by said instruction memory, said register file having outputs selectively in communication said memory functional unit, said register file comprised of input registers and local registers, said local registers in communication with output from said memory functional unit.

11. (Previously Presented) The method of claim 9 wherein said general purpose processor comprises a central processing unit that executes computer instructions.

12. (Previously Presented) The method of claim 9 wherein said logic processors comprise:

an instruction memory;

a logic functional unit that executes Boolean logic instructions; and

a register file controlled by said instruction memory, said register file having outputs selectively in communication with said logic functional unit, said register file comprised of input registers and local registers, said local registers in communication with output from said logic functional unit.

13. (Previously Presented) The method of claim 9 wherein said macro processors comprise:

an instruction memory;

a macro processor executes macro instructions; and

a register file controlled by said instruction memory, said register file having outputs selectively in communication said macro functional unit, said register file comprised of input registers and local registers, said local registers in communication with output from said macro functional unit.

14. (Original) The method of claim 8 wherein said partitioning step comprises:

consulting a programmable logic device library that has a preprogrammed mix of said processor types; and

selecting an appropriate preprogrammed mix of said processor types for said operations for execution.

15. (Original) The method of claim 8 wherein said scheduling step comprises:

creating a program for instruction memories within each of said processor types; and

creating programming files for each programmable logic device used for verifying the user design.

16. (Original) The method of claim 15 further comprising loading said program into each of said instruction memories.

17. (Currently Amended) A method for verifying functionality of an electronic design, the electronic design including Boolean logic gates, at least one macro function and at least one memory circuit, comprising;

compiling the electronic design into logic processors that execute the Boolean logic gates, at least one macro processor that executes the at least one macro function, at least one memory processor that executes the at least one memory circuit, and an interconnect architecture that interconnects said logic processors, said at least one macro processor and said at least one memory processor to one another;

programming said logic processors, said at least one macro processor and said at least one memory processor into at least one programmable logic device such that said at least one programmable logic is programmed to implement said logic processors, said at least one macro processor and said at least one memory processor;

applying stimulus to said logic processors programmed into said at least one programmable logic device, said at least one macro processor programmed into said at least one programmable logic device and said at least one memory processor programmed into said at least one programmable logic device such that said logic processors execute the Boolean logic gates, said at least one macro processor executes the at least one macro function and said at least one memory processor executes the at least one memory circuit; and

collecting output responses generated by said logic processors programmed into said at least one programmable logic device, said at least one macro processor programmed into said at least one programmable logic device and said at least one memory processor programmed into said at least one programmable logic device.

18. (Previously Presented) The method of claim 17 wherein said interconnect architecture further comprises:

an instruction memory;

a plurality of buffers, wherein the number of said plurality buffers is equal to the sum of the number of said logic processors added to the number of said at least one macro processor added to the number of at least one said memory processor, each of said plurality of buffers having an output that is selected by said instruction memory;

a plurality of selectors, wherein the number of said plurality of selectors is equal to the number of said plurality of buffers, each of said plurality of selectors in communication with each of said plurality of buffers so that data stored in any of said plurality of buffers can be transmitted to any of said plurality of selectors, each of said plurality of selectors controlled by said instruction memory; and

a plurality of output ports, each of said plurality of output ports corresponding to one of said plurality of selectors.

19. (Previously Presented) The method of claim 17 wherein said at least one memory processor comprises:

an instruction memory;

a register file controlled by said instruction memory, said register file having outputs selectively in communication said memory functional unit, said register file comprised of input registers and local registers, said input registers in communication with said interconnect architecture, said local registers in communication with output from said memory functional unit.

20. (Previously Presented) The method of claim 17 further comprising a general purpose processor, said general purpose processor comprising a central processing unit that executes computer instructions, said general purpose processor in communication with said logic processors and said at least one macro processor through said interconnect architecture.

21. (Previously Presented) The method of claim 17 wherein said logic processors comprise:

an instruction memory;

a register file controlled by said instruction memory, said register file having outputs selectively in communication said logic functional unit, said register file comprised of input registers and local registers, said input registers in communication with said interconnect architecture, said local registers in communication with output from said logic functional unit.

22. (Previously Presented) The method of claim 17 wherein said at least one macro processor comprises:

an instruction memory; and

a register file controlled by said instruction memory, said register file having outputs selectively in communication said macro functional unit, said register file comprised of input registers and local registers, said input registers in communication with said interconnect architecture, said local registers in communication with output from said macro functional unit.

23. (Cancelled)

REMARKS

Claims 8 and 17 have been amended. Claims 1-7 and 23 have been cancelled without prejudice to focus on method claims 8-22. Claims 8-22 are pending in the application. Applicant respectfully requests reconsideration.

Claim Objections

Claims 8-22 were objected to because of the following informalities:

It is not clear what the Applicant intend to mean by claiming "processor programmed **into** said programmable logic device."

Independent claim 8 has been amended to recite "programming said processor types into said at least one programmable logic device such that said at least one programmable logic device is programmed to implement said processor types" (emphasis added). Claim 8, as amended, makes clear that the programmable logic device is programmed to implement said processor types. Therefore, Applicant submits that the meaning of "programming said processor types into said at least one programmable logic device" is clear, and the objection has been overcome.

Independent claim 17 has been amended in a similar manner as claim 8, and therefore overcomes the objection for the reasons given above.

Claim Rejections under 35 U.S.C. § 102

Claims 8-22 were rejected under 35 U.S.C. § 102(e) as being anticipated by Hoare et al. (U.S. Patent Application Publication 20020133325). Applicant respectfully traverses.

Claim 8 is patentable because Hoare fails to disclose, teach or suggest "mapping the user design into operations for execution; partitioning each of said operations into processor types suitable for each of said operations" **and** "programming said processor types into said at least one programmable logic device such that said at least one programmable logic device is programmed to implement said processor types." (emphasis added).

Hoare discloses a discrete event simulator for simulating a circuit comprising four different types of simulation engines: a logic simulation engine, a memory simulation engine, a behavioral simulation engine, and a interconnection and net-list/annotation simulation engine (Hoare, paragraphs [0111], [0112], [0115], and [0117]). The simulator also comprises an event scheduler for scheduling events among the different simulation engines. To simulate a circuit, the circuit is partitioned into four functionally different types of logic representations (gate logic, memory, behavioral, and interconnections), which are mapped to the different types of simulation engines to perform the simulation (Hoare, paragraph [0123]).

However, nowhere does Hoare disclose, teach or suggest programming the processors of the simulation engines into a programmable logic device such that the programmable logic device is programmed to implement the processors of the simulation engines[1]. While discussing the use of programmable logic devices in the context of hardware logic emulators (Hoare, paragraphs [0080], [0102] and [0103]), Hoare does not disclose using a programmable logic device in the discrete event simulator, much less programming the processors of the simulation engines into a

1 The Office Action relies on the processors within the simulation engines as disclosing the processor types.

programmable logic device. If anything, Hoare teaches away from programming the processors of the simulation engines into a programmable logic device. This is because Hoare states that "hardware emulators can only emulate, not simulate, and they lack the functionality of correctly simulating the circuit's characteristics given by the designer's intention and/or target technology." (emphasis added) (Hoare, paragraph [0103]). Because the purpose of the simulation engines is to simulate and Hoare states that hardware emulators lack the functionality to correctly simulate, Hoare teaches away from programming the processors of the simulation engines into a programmable logic device. Since Hoare does not disclose, teach or suggest programming the processors of the simulation engines into a programmable logic device, the simulation engines of Hoare cannot possibly disclose partitioning the operations of a circuit design into processor types **and** "programming the processor types into a programmable logic device such that the programmable logic device is programmed to implement the processor types," as required by claim 8.

By programming the processor types into a programmable logic device instead of using dedicated (fixed) processors, the method of claim 8 enables a verification system to select the processor types best suited for verifying the operations of a particular user design and to program the selected processor types into the programmable logic device. Hoare does not disclose, teach or suggest programming the processors of the simulation engines into a programmable logic device, and therefore does not provide this advantage. In fact, Hoare discloses the logic simulation engine using a "full custom architecture", which is consistent with dedicated processors (Hoare, paragraph [0111]). Further, Hoare teaches away from programming the

processors of the simulation engines into programmable logic devices as explained above.

Claim 8 is also not disclosed, taught or suggested by the hardware logic emulator discussed in Hoare, in which a circuit design is transformed into a gate level design, instead of processor types, and the gate level design is mapped into a programmable logic device (Hoare, paragraphs [0079] and [0080]).

For at least the reasons given above, Applicant submits that independent claim 8 is patentable over Hoare, and respectfully requests that the rejection of claim 8 be withdrawn.

Claims 9-16 depend from claim 8, and are therefore patentable for at least the reasons given for claim 8.

Claim 17 is patentable because Hoare fails to disclose, teach or suggest compiling an electronic design into logic processors, a macro processor, and a memory processor and programming the logic processors, macro processor and memory processor into a programmable logic device such that the programmable logic is programmed to implement these processors, as required by claim 17. As explained above with regard to claim 8, Hoare does not disclose, teach or suggest programming the processors of the simulation engines into a programmable logic device. Therefore, Applicant submits that claim 17 is patentable over Hoare, and respectfully requests that the rejection of claim 17 be withdrawn.

Claims 18-22 depend from claim 17, and are therefore patentable for at least the reasons given for claim 17.

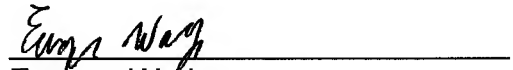
Conclusion

For at least the reasons set forth above, it is submitted that claims 8-22 are in condition for allowance. A Notice of Allowance is earnestly solicited. The Examiner is encouraged to contact the undersigned at (949) 567-6700 if there is any way to expedite the prosecution of the present application.

Respectfully submitted,

Orrick, Herrington & Sutcliffe LLP

Dated: August 18, 2006

By: 
Eugene Worley
Reg. No. 47,186

Orrick, Herrington & Sutcliffe LLP
4 Park Plaza, Suite 1600
Irvine, California 92614-2558
Telephone: (949) 567-6700
Facsimile: (949) 567-6710